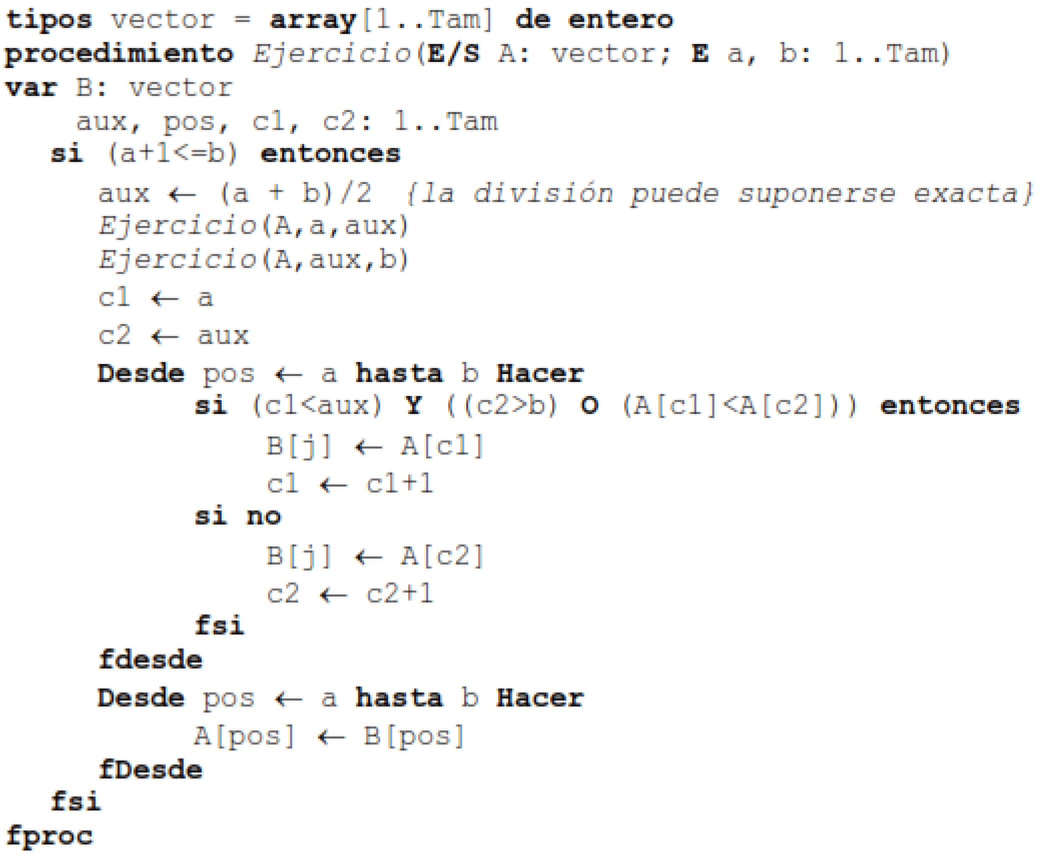
Problems: Chapter 1.

1) Solve the following recursive equations:

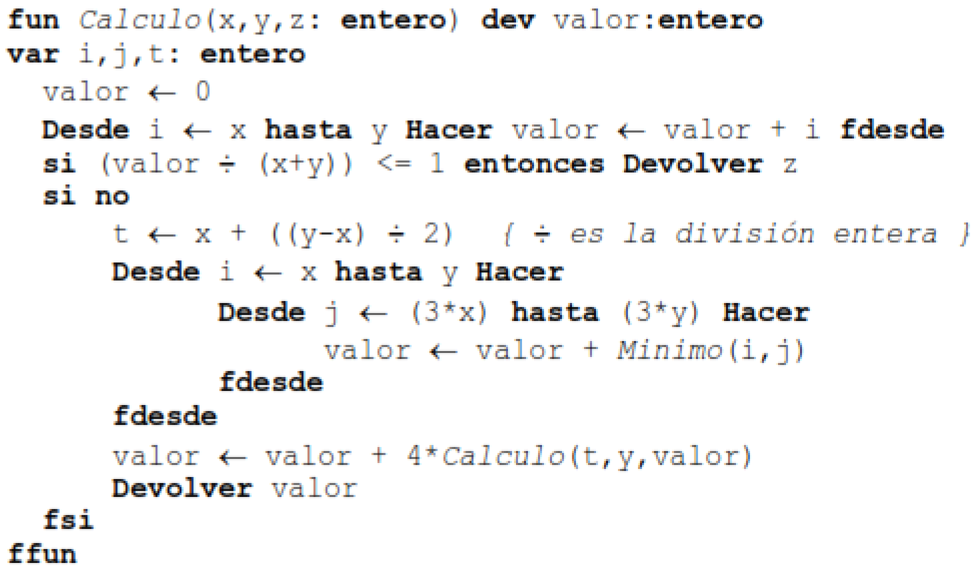
a) $T(n)=2T(n-1)-T(n-2)+8$
b) $T(n)=4n+3+T(n-1)+2T(n-2)$
c) $T(n)=2n+1+2T(n/3)$
d) $T(n)=T(n/2)+n^2+2n$

2) Analyze the efficiency of the following code:



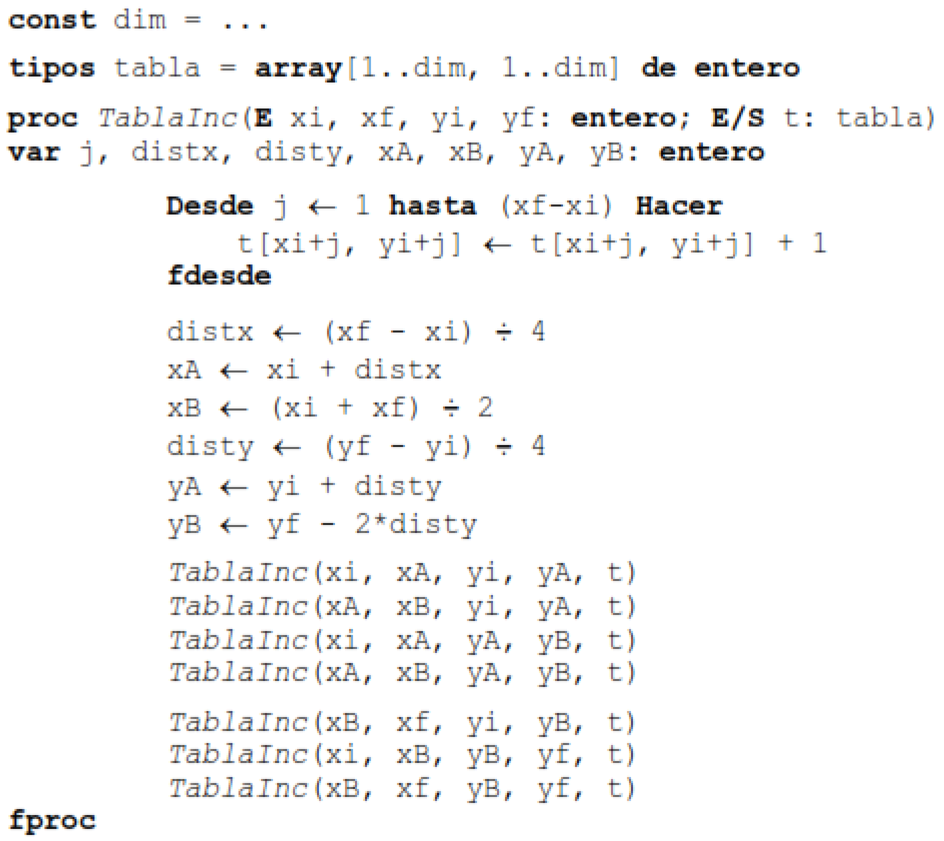
```
tipos vector = array[1..Tam] de entero
procedimiento Ejercicio(E/S A: vector; E a, b: 1..Tam)
var B: vector
    aux, pos, c1, c2: 1..Tam
   si (a+1<=b) entonces
      aux ← (a + b)/2  {la división puede suponerse exacta}
      Ejercicio(A,a,aux)
      Ejercicio(A,aux,b)
      c1 ← a
      c2 ← aux
      Desde pos ← a hasta b Hacer
            si (c1<aux) Y ((c2>b) O (A[c1]<A[c2])) entonces
                 B[j] ← A[c1]
                 c1 ← c1+1
            si no
                 B[j] ← A[c2]
                 c2 ← c2+1
            fsi
      fdesde
      Desde pos ← a hasta b Hacer
            A[pos] ← B[pos]
      fDesde
   fsi
fproc
```

3) Analyze the efficiency of the following code:



```
fun Calculo(x,y,z: entero) dev valor:entero
var i,j,t: entero
  valor ← 0
  Desde i ← x hasta y Hacer valor ← valor + i fdesde
  si (valor ÷ (x+y)) <= 1 entonces Devolver z
  si no
      t ← x + ((y-x) ÷ 2)   { ÷ es la división entera }
      Desde i ← x hasta y Hacer
             Desde j ← (3*x) hasta (3*y) Hacer
                    valor ← valor + Minimo(i,j)
             fdesde
      fdesde
      valor ← valor + 4*Calculo(t,y,valor)
      Devolver valor
  fsi
ffun
```

4) Analyze the efficiency of the following code:



```
const dim = ...
tipos tabla = array[1..dim, 1..dim] de entero
proc TablaInc(E xi, xf, yi, yf: entero; E/S t: tabla)
var j, distx, disty, xA, xB, yA, yB: entero
         Desde j ← 1 hasta (xf-xi) Hacer
              t[xi+j, yi+j] ← t[xi+j, yi+j] + 1
         fdesde

         distx ← (xf - xi) ÷ 4
         xA ← xi + distx
         xB ← (xi + xf) ÷ 2
         disty ← (yf - yi) ÷ 4
         yA ← yi + disty
         yB ← yf - 2*disty

         TablaInc(xi, xA, yi, yA, t)
         TablaInc(xA, xB, yi, yA, t)
         TablaInc(xi, xA, yA, yB, t)
         TablaInc(xA, xB, yA, yB, t)

         TablaInc(xB, xf, yi, yB, t)
         TablaInc(xi, xB, yB, yf, t)
         TablaInc(xB, xf, yB, yf, t)
fproc
```

5) Program a function to determine if a number received as parameter is prime. Analyze the efficiency and complexity.

6) Program a function to determine if a number received as parameter is perfect. Analyze the efficiency and complexity.

7) Write a program which ask a positive number to the user (N) and obtain how many prime numbers there are between 1 and that number N, and how many perfects between 1 and N. Analyze the efficiency and complexity.

8) Program a recursive procedure to obtain the inverse number of a given one.
Example : 627 -> 726
Analyze the efficiency and complexity.

9) Program a recursive function to calculate the following sum:
S= 1+2+3+4+….+n-1+n.
Analyze the efficiency and complexity.

Deliveries: Problem 3, a problem to choose among problems 5, 6 and 7 and a problem among problems 8 and 9.